For this and all future assignments, use the default memory map for the microcontroller, in which program code begins in flash memory at address 0x08000000 and data begins in SRAM at address 0x20000000. Note that the program counter will need to be initialized to the address of the first byte of program code (0x08000000). This setup will enable the simulator and the Discovery board to be used interchangeably.

1. In the Arm program on the next page, a block of “constant data” has been defined, which will be placed by the assembler in the CODE area in memory, following the codes for the instructions. With program code beginning at address 0x08000000, assume that the first constant data value will be placed by the assembler at address 0x08000028.
   - Assembler directives “dcd”, “dcb”, and “space” are described in section 3.6 of the text.
   - Program instructions are described briefly in section 3.5.
   - Instructions “add” and “sub” are described in section 3.4
   - Instructions “ldr” and “mov” are described in sections 5.1 and 5.3.

2. Determine the memory contents allocated by the assembler for the constant data part of the test program (locations pp through tmp), beginning at address 0x08000028. In each row of the table, enter the address of that byte of memory and the value that will be stored in that byte. Also, next to the appropriate rows, indicate which byte of memory corresponds to each symbolic label. Use hexadecimal form for each number.

   **Example:**
   - Lab1 dcd 0x11223344
   - Lab2 dcb 1,-1

<table>
<thead>
<tr>
<th>Label</th>
<th>Memory byte address</th>
<th>Hex value in the byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lab1</td>
<td>0x08000028</td>
<td>44 (Least Significant Byte)</td>
</tr>
<tr>
<td></td>
<td>0x08000029</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>0x0800002A</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>0x0800002B</td>
<td>11 (Most Significant Byte)</td>
</tr>
<tr>
<td>Lab2</td>
<td>0x0800002C</td>
<td>01</td>
</tr>
<tr>
<td></td>
<td>0x0800002D</td>
<td>FF</td>
</tr>
<tr>
<td>....</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. For each of the CPU instructions in the program, determine the 32-bit hex value that you predict would be loaded into the indicated register as the program executes, and enter that value on the line to the right of the instructions.

4. Create a new ARM project, enter the program, assemble it, load it into the simulator with the debugger.
   a. Use the Memory window to check your answers for Step 2 above.
   b. Single-step through the program in the debugger to check your answers from Step 3 above. Next to your answers, record the register values observed in the debugger, and comment on any values that are different from what you found in part 3.

5. Repeat step 4, but this time load the program into the memory of the actual microcontroller on the Discovery board.
Submit the table from Step 2 and a printout of the program with your values from parts 3 and 5 entered on the lines provided.

; Test program
AREA  RESET, CODE
THUMB
ENTRY
Main

; Register contents after instruction:

; (3) My prediction (5) Actual value

mov r0,#10 ; r0 =_________ __________
ldr r1,=pp ; r1 =_________ __________
ldr r2,[r1] ; r2 =_________ __________
mov r1,r0 ; r1 =_________ __________
add r3,r2,r0 ; r3 =_________ __________
ldr r1,=fr ; r1 =_________ __________
ldr r2,=ir ; r2 =_________ __________
ldr r3,[r1] ; r3 =_________ __________
ldr r4,=tmp ; r4 =_________ __________
ldr r5,[r2] ; r5 =_________ __________
ldr r1,=512 ; r1 =_________ __________
add r2,r1,r5 ; r2 =_________ __________
sub r0,r6,r6 ; r0 =_________ __________

Here b Here ; Effectively “halts” the program

; Constant data defined in code memory - these values follow the code for the last instruction
pp dcd 0x12345678
ir dcb 5,6,7,8
ts dcd 0x0805
fr dcb 'A'
reg dcd 18,-20,15,-10
mem space 8
tmp dcd 0x1234,0x5678

; No DATA area for this program – above data is all in the CODE area

END